GENERATING COHERENT ARGUMENTATIVE PARAGRAPHS

Michael Elhadad
Department of Computer Science
450 Computer Science Building
Columbia University
New York, N.Y. 10027
ELHADAD@CS.COLUMBIA.EDU[1]

Abstract

We address the problem of generating a coherent paragraph presenting arguments for a conclusion in a text generation system. Existing text planning techniques are not appropriate for this task for two main reasons: they do not explain how *arguments* can be linked together in a linear presentation order and they do not explain how the rhetorical function of a proposition affects its wording.

We present a mechanism to generate argumentative paragraphs where argumentative relations constrain not only the rhetorical structure of the paragraph, but also the surface form of each proposition. In our approach, a text planner relies on a set of specific argumentative relations to extract information from the knowledge base, to map it to scalar and context dependent evaluations and to organize it into chains of arguments. The same information used for planning is also used by the surface realization component to perform lexical choice at all the levels of the clause (connectives, main verb, adverbial adjuncts, adjectives and determiners). The mechanism is implemented in the ADVISOR II system using FUF, an extended functional unification formalism.

INTRODUCTION: MOTIVATION

Certain types of questions require in response a statement of a conclusion and arguments to support it. In our domain, a question-answering system offering advice to students selecting classes to plan their schedule (McKeown, 1988), should-questions, *e.g., should I take AI?*, fall into this class. The example shown in Figure 1, extracted from a corpus of naturally occurring advising sessions that we have collected, illustrates this point.

The task we consider is that of generating similar argumentative paragraphs presenting an evaluation of a course and its supporting arguments. To produce such paragraphs, a generation system must determine which arguments to include in the paragraph, how to organize them in a structured paragraph, and how to phrase each piece of the argument. For example in Figure 1, the advisor selects the argument chain that AI provides preparation for all followup courses in the field, that the more the student is interested in AI the more he should take these followup courses and therefore, the more

```
Q Should I take AI this semester?

A If you want to take courses like
  Natural Language Processing or
  Expert Systems or Vision
  next semester,
  it's very advisable you take AI
    because
    that's going to help you a lot
  So if you are interested
  in the whole field at all,
  I would advise you strongly
  to take AI now.
```

Figure 1: An argumentative paragraph

reason he has to take AI. This sequence of arguments forms the structure of the answer.

In terms of wording, note that the conclusion that is supported affects the choice of expressions at many levels. We have marked in italics words that are selected in part because of the argumentative function of the proposition in which they appear. For example, saying *it is very advisable* as opposed to *it is OK*, deciding to add *strongly* and selecting *a lot* instead of *somewhat* are all decisions motivated by the advisor's goal of convincing the student to take AI.

In previous work in text generation, rhetorical schemas (McKeown, 1985) and RST (rhetorical structure theory) (Mann & Thompson, 1987) have been proposed as operational techniques to produce coherent paragraphs. We have found, however, that these techniques, in their current forms, are not appropriate to address the task of generating *argumentative* paragraphs for two main reasons: first, RST relations are too generic to perform argument selection and construct coherent argument chains; second, rhetorical relations in both theories do not influence directly linguistic realization and therefore cannot determine wording decisions of the type illustrated in Figure 1.

We present in this paper a mechanism for planning and realizing argumentative paragraphs which addresses these two shortcomings. In our approach, specific argumentative relations guide both content planning and lexical choice within the clause. Content planning is

[1]Reprinted from Proceedings of COLING'92, August 1992, Nantes, France.

performed using two levels of argumentative relations - evaluation functions and topoi (Anscombre & Ducrot, 1983) to derive content from the underlying knowledge base and organize it into coherent argumentative chains. Surface realization takes advantage of the output of the paragraph structurer to perform lexical choice at all levels of the clause.

In the rest of the paper, we first review previous work in paragraph planning, explaining why existing techniques cannot be used directly in the case of argumentative paragraphs. We then present our approach, describing the content planner and the surface realization component.

PREVIOUS WORK: SCHEMAS AND RST

In previous work in text generation, two methods have emerged to generate coherent paragraph-long texts: rhetorical schemas and RST (for Rhetorical Structure Theory).

Schemas (McKeown, 1985) encode conventional patterns of text structure. A schema is associated with a communicative goal and describes how this goal is conventionally satisfied. For example, the constituency schema is used to describe the parts of an object, and the process schema (Paris, 1987) is used to describe a complex process. A schema describes a sequence of *rhetorical predicates* where each predicate is either a primitive communicative function, which can be fulfilled by a single proposition, or recursively another schema. For example the primitive predicate *attributive* attributes a property to an object. Each predicate is assigned a semantics in terms of a query to a knowledge base, therefore when the schema is traversed, propositions are retrieved from the knowledge base as predicates are instantiated. The output of a schema traversal is therefore a sequence of propositions labeled by the name of the rhetorical predicate they instantiate.

While schemas label each proposition as the instantiation of a predicate, RST attempts to label the *relation* between propositions. RST (Mann & Thompson, 1987) was first introduced as a descriptive theory aiming at enumerating possible rhetorical relations between discourse segments. RST relations include *elaboration*, *anti-thesis*, *evidence* and *solutionhood*. A relation connects two text spans, which can be either single propositions or recursively embedded rhetorical relations. One argument of the relation is marked as its ''nucleus'' while the others are the ''satellites'' and are all optional.

RST was made operational as a technique for planning the structure of paragraphs in (Hovy, 1988a) and (Moore & Paris, 1989). The idea is to attach a communicative intent with each RST relation and to view the combining of relations into paragraphs as a planning process, decomposing a high-level intention into lower-level goals that eventually can be mapped to single propositions. The communicative goals associated with the leaves of the structure are then used to retrieve the content of each proposition from an underlying knowledge base. By making the intentional structure of a paragraph explicit, this work follows the discourse structure theory advanced in (Grosz & Sidner, 1986). Note also that, since in RST with planning, the structure of paragraphs is dynamically derived, it is possible to view schemas as the compilation of RST configurations with some information abstracted out, as pointed out in (Mann, 1987).

We found that schemas and RST were not appropriate for planning and generating argumentative paragraphs because argument selection cannot be easily performed. Among the types of relations enumerated in RST, only two would apply to the analysis of argumentative paragraphs: *evidence* and *thesis-antithesis*. If these relations were to be composed into a paragraph structure, they would yield a chain of undistinguished *evidence* links. To determine which propositions can serve as arguments and how to order them, one needs to specify precisely how arguments in the domain combine and relate to a conclusion. An RST type of approach cannot be used alone to plan the content of an argumentative paragraph. Schemas suffer from the same limitation.

In place of a generic relation like *evidence*, we use specific argumentative relations called topoi (Anscombre & Ducrot, 1983), *e.g., the more a class is difficult, the less a student wants to take it*, to perform content selection. The mechanism is detailled later in the paper.

Rhetorical Relations and Lexical Choice

While rhetorical schemas or RST have been used to determine the content of the paragraph and the ordering of the propositions, they have not been used to determine the surface form of the clause. We have found, however, that in argumentative paragraphs, the rhetorical function of a proposition affects its wording at many levels. Consider the following utterances, extracted from our corpus:

(1) It requires quite a lot of programming

(2) It does involve some programming, but nothing outrageous.

Our contention is that either (1) or (2) can be generated from the *same* content as input, but that the difference between the two forms is determined by the argumentative function of the clause: (1) supports the conclusion that a course should not be taken because it requires a lot of programming, which is time consuming and therefore makes the course difficult. In contrast, (2) supports the conclusion that the level of programming should not affect the decision whether to take the course.

The amount of programming involved in a course can be quantified by considering how many programming assignments are required and the number of programming projects. The question is then, given this information, how to describe this information to a student: what level constitutes *some programming*, *quite a lot of*

programming or *a not outrageous amount of programming*?

Our position is that the mapping from the objective information that a course requires two programming assignments to an evaluation that it requires *some programming* is only partially determined by the content. It is also and over all a rhetorical decision. It is because we want to support a certain conclusion that we view and evaluate an objective quantity as *a lot* or *some*.

In addition, by looking back at examples (1) and (2), we find that this rhetorical decision also affects the choice of the main verb: the course *requires* programming when the evaluation of the course is negative, while it *involves* programming when the evaluation is positive. In (Hovy, 1988b), similar issues of lexical choice were also addressed, but different mechanisms were used to perform lexical choice and paragraph organization.

This is an instance of the general problem of expressibility discussed in (Meteer, 1990): RST and schemas in their current form do not bridge the gap between rhetorical relations and surface realization, and as a consequence, surface realization cannot take advantage of the paragraph organization to make decisions.

In earlier work, we have studied the problem of generating certain connectives like *but, although, because* or *since* (Elhadad & McKeown, 1990) and of generating adjectives (Elhadad, 1991). In both cases, we have found that argumentative features play an important role in the selection of appropriate wording. The important point, is that the same argumentative features could be used to constrain both the choice of connectives *between* the clause and the choice of adjectives *within* the clause. The particular argumentative features we use are inspired from work by (Anscombre & Ducrot, 1983), (Bruxelles *et al*, 1989) and (Bruxelles & Raccah, 1991). In this paper, we show how these argumentative features can be generated by a paragraph structurer, and therefore serve as a bridge between the rhetorical function of a clause and its surface realization.

OUR APPROACH

In order to explain how lexical choice within the clause can be affected by the rhetorical function of a proposition, we must design a text planner that annotates the propositions with information about their argumentative function. In the ADVISOR system, the following activities are performed to produce the answer to a should-type question:

1. An expert-system determines whether the course should be taken.
2. An evaluation system maps observations about the course from the knowledge base into evaluations that are scalar and context-dependent.
3. The evaluation system links these evaluations into argument chains using argumentative relations called topoi. Topoi relations are stored within the propositions as a separate feature.
4. A paragraph structurer selects and organizes argumentative chains into an argumentative strategy.
5. A surface realization component maps the argumentative strategy into a paragraph, relying on a grammar which is sensitive to the argumentative information stored in the propositions.

An important feature of this approach is that the mapping between information in the knowledge base and the content of the propositions is performed in two stages by two types of argumentative relations: evaluation functions and topoi. We distinguish between evaluation, which is the leap from the observation of an objective fact in the knowledge base to a context-dependent scalar evaluation, and argumentative relations, which only operate on scalar evaluations, and not on knowledge-base facts. In contrast, most other text planners simply organize propositions *directly* retrieved from the knowledge base.

Another important feature is that we do not use generic rhetorical relations like ‘‘anti-thesis’’ or ‘‘evidence’’ but instead specific argumentative relations called topoi. Because topoi are gradual inference rules, our content planner performs a task similar to generating explanations for a rule-based expert system (McKeown & Swartout, 1987). But in addition to determining content, topoi are also used to influence wording: they are added as annotations to the propositions generated by the text planner and are used by the surface realization component to perform lexical choice.

In the following sections, we detail how content planning is performed and how the grammar takes advantage of the argumentative information placed in its input to perform lexical choice.

CONTENT PLANNING

Our system determines which content can be used to generate an answer in two stages using first evaluation functions then topoi.

Evaluation Functions

Evaluation functions are used to map from observations of facts in the knowledge base to context-dependent evaluations. They are domain specific and rely on the presence of a user-model. An evaluation is the rating of a knowledge-base entity on a scale. In the ADVISOR domain we have identified the relevant scales by examining a corpus of transcripts of advising sessions. We looked at all the adjectives modifying a class in these transcripts and classified them into semantic categories. The following classes were thus identified (details on this analysis are provided in (Elhadad,

1991)):

- Goodness
- Importance
- Level
- Difficulty
- Workload
- Domain: programming and mathematical

Note that all of these categories are scalar and therefore define a set of dimensions along which a class can be evaluated. The task of the evaluation component is to rank a course on relevant scales. In the current implementation, ranking is binary so a course can be in three possible states with respect to each scale: not ranked (the scale is not active in the current context), + (the course is high on the scale) or - (the course is low on the scale). In the current state of the program, there is no distinction between degrees (*interesting* vs. *very interesting*).

Ranking is accomplished by using simple rules which determine under which conditions objective facts stored in the knowledge base can become convincing evidence for an evaluation. Figure 2 shows three evaluation rules used in the current system.

```
If   U(user.programming.-) &
     K(class.programming-hw > 0)
then E(class.programming +)

If   U(user.programming.*) &
     K(class.programming-hw = 0)
then E(class.programming -)

If   U(user.programming.+) &
     K(class.programming-hw > 3)
then E(class.programming +)
```

Figure 2: Sample evaluation rules

`U(user.programming -)` checks if in the current state of the user model the system has evidence that the user dislikes programming. `K(class.programming-hw > 0)` is a query to the knowledge base to determine whether the class has some programming assigments. An assertion of the form `E(class.programming +)` is a positive evaluation of the course on the `programming` scale. If none of the rules shown in Figure 2 are activated, the `programming` scale will remain non-activated.

If the first rule is activated, a proposition attributing a number of programming assignments to the class is added to the paragraph being planned. In addition, this content is annotated by an evaluation on the `programming` scale. The output of the evaluation system is therefore a set of propositions annotated by evaluations along each of the activated scales.

Argumentative Relations: Topoi

Once the course has been evaluated on the activated scales, the evaluation system considers relations between the scales. We use the notion of topoi as defined in (Anscombre & Ducrot, 1983) to describe such relations. Topoi are gradual inference rules of the form ‘‘the more/less X is P, the more/less Y is Q.’’ Figure 3 shows sample topoi used in the ADVISOR system.

```
workload + / difficulty +
workload + / time-required +

difficulty + / workload +
difficulty + / time-required +
difficulty + / take -

programming + / time-required +

interest + / take +
importance + / take +
```

Figure 3: Sample topoi used in ADVISOR

Topoi play the role of rhetorical relations in RST by explaining the relation between two propositions in a paragraph. But they are different in that they are very specific relations as opposed to generic relations like ‘‘anti-thesis’’ or ‘‘evidence’’. They can therefore be used to determine the content of the answer and the order in which arguments should be presented.

But the most important feature of topoi for our purposes is that they can be related to lexical choice in a natural way. In (Bruxelles *et al*, 1989) and (Bruxelles & Raccah, 1991) it is suggested that lexical items can be defined in part by their argumentative potential. For example, it is part of the definition of the verb ‘‘to require’’ as used in our domain, that its subject is evaluated on the scale of difficulty. This argumentative connotation explains the contrast between (3) and (4), in a context where both are addressed to a student who enjoys programming:

(3) ? At least AI requires programming, so it’s easy.

(4) At least AI involves programming, so it’s easy.

The same scales are used both in topoi and in our lexical description. They therefore serve as a bridge between the rhetorical structure of the paragraph and lexical choice.

A GRAMMAR SENSITIVE TO ARGUMENTATIVE CONSTRAINTS

The output of the evaluation system is a list of chains of acceptable argumentative derivations supporting the conclusion that a course should be taken or not. Each proposition in the chain is annotated by a feature AO for Argumentative Orientation which indicates how it relates to the surrounding propositions. Figure 4 shows a sample proposition using the notation of functional

```
((cat lex-verb)
 (alt verbal-lexicon (:index concept)
   (((concept c-contain)
     (alt contain-lex (:bk-class ao)
       (
        ;; Verbs argumentatively marked
        (({AO} ((scale s-difficulty)
                (carrier c-class)
                (carrier {participants carrier})
                (orientation +)
                (ao-conveyed lex-verb)))
         ({participants carrier} c-class)
         (lex ((ralt ("require" "demand")))))

        ;; Neutral verbs
        ((lex ((ralt ("contain"
                      "involve")))))))

      <<other concepts>>)))
```

Figure 5: Fragment of the grammar

descriptions (FDs) used in functional unification grammars.

```
((cat relation)
 (name topics-of)
 (roles
  ((class ((cat class)
           (name AI)))
   (topics
    ((cat set)
     (kind ((cat topic)))
     (cardinality 1)
     (intension
      ((cat relation)
       (name area-of)
       (argument {^ roles topic})
       (roles ((topic ((cat topic)))
               (area ((name theory))))
     (extension
      ((cat list)
       (elements ~(((name logic)))))))
 (AO
  ((scope
    ((cat clause)
     (type attributive)
     (participants
      ((carrier {roles class})
       (attribute ((cat scale)))))))
     (scale ((name theoretical)))
     (orientation +)
     (focus
      {^ scope participants carrier})
     (scalar
      {^ scope participants attribute}
     (conclusion [+ difficult(AI)])))))
```

Figure 4: Input to the grammar

This input represents the proposition that AI covers (among others) a set of topics in the area of theory (namely, logic), and the AO feature indicates that this proposition is used as an argument for the conclusion that AI is a difficult course, by virtue of the topos `theoretical + / difficult +` (the conclusion part of the topos is shown in abbreviation). Details on the role of each fields in the AO feature and on the representation of quantification are provided in (Elhadad, 1992).

Because of this AO specification, the grammar will choose appropriately realization (5) instead of (6):

(5) AI requires a lot of programming

(6) AI involves some programming.

The realization component is implemented in FUF, an extended functional unification grammar formalism which we have implemented (Elhadad, 1990, Elhadad, 1992). In the grammar we use, lexical choice and syntactic realization are interleaved. For example, the choice of the verb is handled by the alternation shown in Figure 5. In this Figure, the notation `alt` indicates a disjunction between alternatives; `ralt` indicates a random alternation, and is used to indicate that the grammar does not account for the difference between the alternatives; the curly braces notation in pairs of the form `({AO} value)` indicates that the `AO` feature is not embedded in the lexical verb constituent unified with the grammar but rather is a top level feature within the clause.

The fragment shown in Figure 5 specifies how the grammar can map from an input concept `c-contain` to a verb expressing this relation. The grammar for this relation contains two branches: in the first branch, the verbs ‘‘require’’ and ‘‘demand’’ are described as being argumentatively marked on the scale of difficulty. They can therefore be selected to project an evaluation on their subject. Note that the choice between ‘‘require’’ and ‘‘demand’’ is arbitrary, as indicated by the `ralt` construct - it is not explained by the grammar. The second branch describes the verbs ‘‘contain’’ and ‘‘involve’’ as neutral verbs, that do not add any connotation.

When there is an argumentative connotation, the grammar specifies which participant in the clause is affected by the argumentative evaluation (for both verbs in the example, the subject of the verb is the entity that carries the evaluation). Similar lexical descriptions for adjectives are described in (Elhadad, 1991).

The part of the grammar generating the syntactic structure of the clause is inspired both by systemic grammars ((Halliday, 1985) and especially (Fawcett, 1987)) for the semantic features of the input and by HPSG (Pollard & Sag, 1987) for the overall flow of control. It has been extended to account for the flow of argumentative information from lexical items to constituents and to the clause. For example, inserting an adjective argumentatively marked as the describer of a noun group creates an argumentative orientation feature at the level of the noun group which is then percolated to the clause in which the noun group is a participant.

Finally, the clause grammar has been extended with a clause complex constituent which determines connective selection and clause combining (an extension of (McKeown & Elhadad, 1991)). A clause complex is represented as an FD with features `directive` and `subordinate` (a notion similar to the RST distinction between nucleus and satellite). As discussed in (Elhadad & McKeown, 1990), there are many different connectives expressing argumentative relations. For example all of the following connectives can be used to express an evidence relation: *because, since, therefore, so, as a consequence, then*. The choice offered to the analyst is then: (i) to ignore the differences between such close connectives; (ii) to define a single rhetorical relation for each connective or (iii) to determine the choice of connective on other factors than the rhetorical relation alone. We adopt this later approach, and conclude that the output of the paragraph structurer must not determine the connectives, as is generally done by schema or RST based planners. Instead we take advantage of how our text planner labels each proposition with information about its rhetorical function to determine which connective is most appropriate in combination with the other pragmatic factors discussed in (Elhadad & McKeown, 1990). In this paper, we have also explained how the argumentative features needed to select connectives are produced by the content planner.

Implementation

The content planner is fully implemented. In the surface realization component, the clause grammar is fully implemented with account for argumentative features in adjectives, verbs and adverbial adjuncts. A large portion of the grammar covers the determiner sequence and how the choice of determiners like ‘‘many’’, ‘‘most’’, ‘‘few’’ etc. has an influence on the argumentative orientation of the clause. The grammar for connectives is separately implemented but not yet merged with the rest of the grammar.

The grammar is quite large: the current version includes 580 disjunctions; it covers simple and complex clauses, interrogation, negation, a complex tense system, relative clauses, control and raising, coordination with some forms of ellipsis. We have extended FUF by adding sophisticated control devices (Elhadad & Robin, 1992), making it possible to handle such large grammars. In particular, we are able to deal with the non-local constraints across constituent boundaries imposed by argumentative relations in an efficient way.

User Profile:
```
Programming  +
Math         -
Year         Soph
Interests    AI, NLP
```

Class profile (AI):
```
Programming Assignments 3
Paper Assignments       1
Projects    0
Topics:     Logic[Math], NLP[AI]
```

Should I take AI?

AI can be difficult,
because it requires a lot of work
and
it is pretty mathematical,
but it is an interesting course,
because it covers many nlp topics,
and
it offers lots of programming hws.

Figure 6: An argumentative paragraph

Figure 6 shows the type of paragraphs obtained when all the pieces of the surface realization component are put together.

CONCLUSION

We have described a model for planning argumentative paragraphs which can perform content selection and which allows the surface realization component to adapt lexical choice within each clause to the rhetorical function of the clause. The model relies on the fact that the same argumentative relations which can be used as specific rhetorical relations also participate in the lexical description of verbs, adjectives, adverbs and determiners.

Our model also distinguishes between two types of argumentative relations: evaluative functions and topoi. Evaluation function fetch information from the knowledge base and make it scalar and context-dependent, while topoi are purely rhetorical relations that link scalar propositions together according to the argumentative goal of the speaker. This two-stage content retrieval mechanism is in contrast to most existing planners which assemble facts directly retrieved from the knowledge base, and do not transform them according to the pragmatic context (goals of the speaker and user model). The mechanism is implemented using the FUF text generation system.

Some of the open questions we face are:

- Deciding whether to use a connective or not.
- Deciding whether propositions can be left implicit and still be recoverable.
- Combining several independent argumentative chains and deciding how to order the arguments in the combined structure.
- Acquiring the argumentative lexical descriptions we need on a large scale
- Scaling up the text planning mechanism to generate several paragraphs.

In particular, when several independent argumentative chains support the same conclusion, argumentative relations alone cannot determine in which order they must be organized. We are currently investigating whether argumentative strategies similar to RST schemas can be combined with our technique. We are also evaluating how other discourse aspects like topic progression can help in the planning of the paragraph.

Acknowledgments. This work was supported by DARPA under contract N00039-84-C-0165, by NSF Grant IRT-84-51438 and by ONR Grant N00014-89-J-1782. I want to thank Kathy McKeown, Jacques Robin and Frank Z. Smadja for helping me write this paper and Chaya Ochs for helping with the corpus analysis and the implementation of evaluation functions.

REFERENCES

Anscombre, J.C. and O., Ducrot. (1983). *Philosophie et langage*. *L'argumentation dans la langue*. Bruxelles: Pierre Mardaga.

Bruxelles, S. and Raccah P.Y. (1991). Argumentation et Semantique: le parti-pris du lexique. In *Actes du Colloque 'Enonciation et parti-pris'*. Forthcoming.

Bruxelles, S., Carcagno, D. and Fournier, C. (1989). Vers une construction automatique des topoi a partir du lexique. *CC AI - Journal for the integrated study of Artificial Intelligence cognitive science and applied epistemology*, *6*(4), 309-328.

Elhadad, M. (1990). Types in Functional Unification Grammars. *Proceedings of 28th Meeting of the ACL (ACL'90)*. Pittsburgh.

Elhadad, M. (1991). Generating Adjectives to Express the Speaker's Argumentative Intent. *Proceedings of 9th National Conference on Artificial Intelligence (AAAI 91)*. Anaheim.

Elhadad, Michael. (1992). *Using Argumentation to Control Lexical Choice: a Functional Unification-based approach*. Doctoral dissertation, Columbia University.

Elhadad, M. and K.R. McKeown. (1990). Generating Connectives. *Proceedings of COLING'90 (Volume 3)*. Helsinki, Finland.

Elhadad, M. & Robin, J. (1992). Controlling Content Realization with Functional Unification Grammars. In R. Dale, E. Hovy, D. Roesner and O. Stock (Ed.), *Aspects of Automated Natural Language Generation*. Springer Verlag.

Fawcett, R.P. (1987). The semantics of clause and verb for relational processes in English. In Halliday, M.A.K. & Fawcett, R.P. (Ed.), *New developments in systemic linguistics*. London and New York: Frances Pinter.

Grosz, B. and Sidner, C. (1986). Attentions, intentions, and the structure of discourse. *Computational Linguistics*, *12*(3), 175-204.

Halliday, M.A.K. (1985). *An Introduction to Functional Grammar*. London: Edward Arnold.

Hovy, E.H. (June 1988). Planning Coherent Multisentential Text. *Proceedings of the 26th ACL Conference*. Buffalo: ACL.

Hovy, E.H. (1988). *Generating natural language under pragmatic constraints*. Hillsdale, N.J.: L. Erlbaum Associates. Based on the author's thesis (doctoral--Yale University, 1987).

Mann, W.C. (1987). *Text generation: the problem of text structure* (Tech. Rep. ISI/RR-87-181). Marina del Rey, CA: ISI.

Mann, W.C. and S.A. Thompson. (1987). Rhetorical Structure Theory: Description and Construction of Text Structures. In Gerard Kempen (Ed.), *Natural Language Generation*. Martinus Nijhoff.

McKeown, K.R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Studies in Natural Language Processing. Cambridge, England: Cambridge University Press.

McKeown, K.R. (1988). Generating Goal Oriented Explanations. *International Journal of Expert Systems*, *1*(4), 377-395.

McKeown, K. and M. Elhadad. (1991). A Contrastive Evaluation of Functional Unification Grammar for Surface Language Generators: A Case Study in Choice of Connectives. In Cecile L. Paris, William R. Swartout and William C. Mann (Eds.), *Natural Language Generation in Artificial Intelligence and Computational Linguistics*. Kluwer Academic Publishers.

McKeown, K.R. & Swartout W.R. (1987). Language generation and explanation. *The Annual Review of Computer Science*, (2), pp. 401-449.

Meteer, M.W. (1990). *The generation gap: the problem of expressibility in text planning*. Doctoral dissertation, University of Massachussets at Ahmerst. Also available as BBN technical report No. 7347.

Moore, J.D. and C.L. Paris. (June 1989). Planning Text for Advisory Dialogues. *Proceeding 27th ACL*.

Vancouver, BC: ACL.

Paris, C.L. (1987). *The Use of Explicit User models in Text Generation: Tailoring to a User's level of expertise*. Doctoral dissertation, Columbia University.

Pollard, C. and I.A. Sag. (1987). *CSLI Lecture Notes*. Vol. 13: *Information-based Syntax and Semantics - Volume 1*. Chicago, Il: University of Chicago Press.